XSEDE Visualization Use Cases

5 March 2014

Version 1.3

Table of Contents

A. Document History

Overall Document Authors:

Kelly Gaither, TACC
kelly@tacc.utexas.edu

Sean Ahern, NICS
ahern@utk.edu

David Bock, NCSA
dbock@ncsa.uiuc.edu

Altaf Hossain, PSC
altaf@psc.edu

Ian Foster
The University of Chicago
and Argonne National
Laboratory
Argonne, IL 60439
foster@anl.gov

Morris Riedel Jülich
Supercomputing Centre
Forschungszentrum Jülich
GmbH
D-52425 Jülich
Germany

Andrew Grimshaw
University of Virginia
PO Box 400740
Charlottesville VA 22904
grimshaw@virginia.edu

David Lifka
Cornell University
512 Frank H. T. Rhodes Hall
Ithaca, NY 14853
lifka@cac.cornell.edu

	Version	Date	Changes	Author
First use case draft	1.3	3/5/2014	Document created	XSEDE Visualization Team

B. Document Scope

This document is both a user-facing document (publically accessible) and an internal working document intended to define user needs and use cases that fall within the overall activities of XSEDE. The definition of use cases is based on a template from Malan and Bredemeyer[1]. In general it is in keeping with the approaches and philosophy outlined in "Software architecture in practice."[2]

This document is one component of a process that generates at least the following documents, some of which are user-facing, some are as of now intended to be internal working documents:

- ***This document*** - A description of use cases [User facing]
- A set of level 3 decomposition documents, which include:
    - Quality Attributes descriptions
    - Connections diagram in UML

The use cases are presented here using the following format, derived from the Malan and Bredemeyer white paper[1] as follows:

Use Case	Use case identifier and reference number and modification history
Description	Goal to be achieved by use case and sources for requirement
References	References and citations relevant to use case
Actors	List of actors involved in use case
Prerequisites (Dependencies) & Assumptions	Conditions that must be true for use case to be possible Conditions that must be true for use case to terminate successfully
Steps	Interactions between actors and system that are necessary to achieve goal
Variations (optional)	Any variations in the steps of a use case
Quality Attributes	
Non-functional (optional)	List of non-functional requirements that the use case must meet
Issues	List of issues that remain to be resolved

[1] Malan, R., and D. Bredemeyer. 2001. Functional requirements and use cases. *www.bredemeyer.com/pdf_files/functreq.pdf*

[2] Bass, L., P Paul Clements, and Rick Kazman

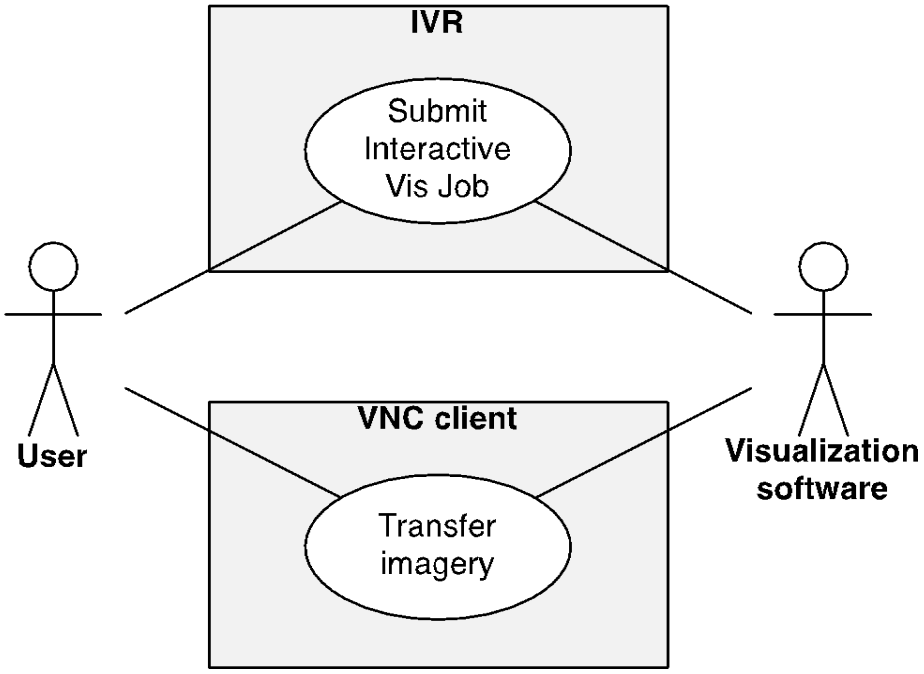
C. Glossary

System: Service within XSEDE that provides HPC capabilities. Accessible across a high-speed network.
OpenGL: A protocol that allows 3D graphics primitives to be turned into imagery.
VNC: A protocol that allows for 2D imagery to be interactively transferred over the network. VNC is serial. It is essentially an X11/OpenGL display. It has both client and server portions.
Frame buffer: A section of memory that serves to contain the color and depth values for each pixel being drawn on the screen. In general, a frame buffer is allocated by a server directly and cannot be externally provisioned. They can be optionally accelerated by co-located GPU resources. For most user needs, a resolution of 1k x 1k is sufficient.
Interactive visualization: Establishment of interactive session with very low (seconds) latency between request for resources and initiation of interactive visualization. Once established, "interactive" is generally recognized as response times between user and software >= 10 frames/second (fps).
VNC Server: The master software that runs on a remote machine and brokers interactive usage (keyboard and mouse events and OpenGL in combination with VirtualGL) by communicating back and forth via TCP/IP with a VNC client running on a user's local machine. It is the program that shares its virtual screen (frame buffer) or desktop.
VNC Client: The program that watches, control and interacts with the VNC server to display a remote desktop/screen on a user's local machine.
In situ visualization: Visualization software/algorithms that run in the same memory space as an HPC simulation.
Frames/sec: A metric for calculating the performance of a visualization. Each frame is a complete image.
Batch: A method of scheduling jobs as resources are available. Does not require that the user be available when the job is run. The opposite of interactive.
IVR: Interactive Visualization Resource (e.g., Gordon, Longhorn, Nautilus). XSEDE resource that is used for data visualization and analysis.

D. Visualization Use Cases

Use Case	**VISUALIZATION.1.0**
Remote, Interactive Visualization (1)	User interacting with a remote system and has need for visualization.
References	
Actors	• XSEDE: Service Provider (SP) Forum Members with remote IVR • XSEDE: visualization teams • XSEDE: documentation and support teams • XSEDE User • Visualization software on the IVR
Prerequisites (Dependencies)	• The resource being used must be a remote IVR.

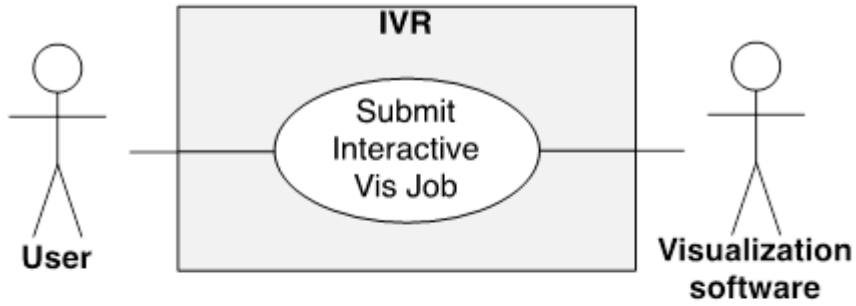
and Assumptions	• The data the user wants to visualize must be accessible by the remote IVR. • The visualization applications that the user wants to use to visualize their data must be written in OpenGL. • The user must use XSEDE authentication and a previously installed VNC client or a java-enabled web browser. • The network being used must provide sufficient bandwidth to provide performance of >= 10 frames/second.
Steps	• User identifies remote IVR that provides pixel delivery • User submits an interactive visualization job to the batch queue on the remote IVR • User gets the display port number from a status file written by the job upon running • User connects to the job running on the remote IVR by connecting to the display port number via VNC running through the client or through a java-enabled web browser
Variations (optional)	
Quality Attributes	• Provide interactive performance of >= 10 frames/second
Non-functional (optional)	
Issues	• The performance of the connection may be highly dependent upon the particular codec used between the job and the VNC client.

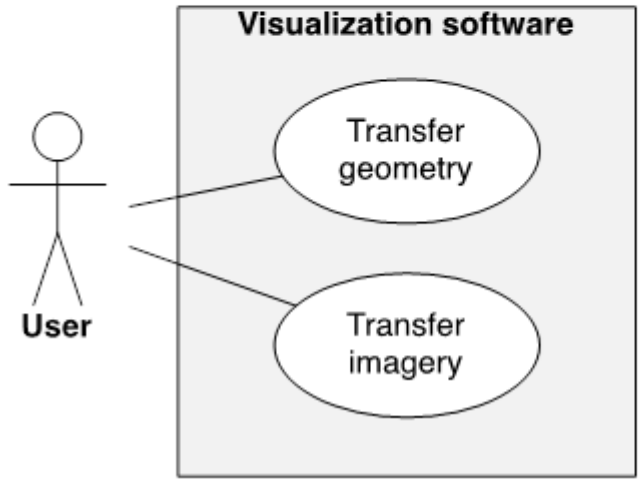


Visualization Use Case 1 Diagram

Use Case	**VISUALIZATION.2.0**
Remote, Interactive Visualization (2)	User interacting with a remote system and has need for visualization.
References	
Actors	• XSEDE: SP Forum Members with remote, interactive visualization resources (IVR) that provide geometry delivery • XSEDE: visualization teams • XSEDE: documentation and support teams • XSEDE User • Visualization software on the IVR
Prerequisites (Dependencies) and Assumptions	• The resource being used must provide software capable of delivering geometry to a user's local resource. • The data the user wants to visualize should be accessible by the resource being used for visualization. • The resource being used for the visualization must have software applications installed that facilitate geometry delivery to the user's local resource. • The user must use XSEDE authentication and the

	user's local machine must have a graphics card capable of performing the local rendering • The network being used for the visualization must have sufficient bandwidth to provide performance of >= 10 frames/second.
Steps	• User identifies remote IVR with visualization software that delivers 3D geometry • User installs any client-side software required by the applications • User submits an interactive visualization job to the batch queue on the remote IVR • User connects to the running batch job via the client-side software
Variations (optional)	• MUST support client software on a user-controlled laptop or desktop, perhaps behind a NAT or firewall. ○ Currently TCP/IP sockets are used - and a TCP/IP compatible implementation is highly desired.
Quality Attributes	• Provide interactive performance of >= 10 frames/second
Non-functional (optional)	
Issues	• The interactive visualization job and the client-side software may negotiate to adjust the data payload, moving between geometry and imagery as the performance dictates.

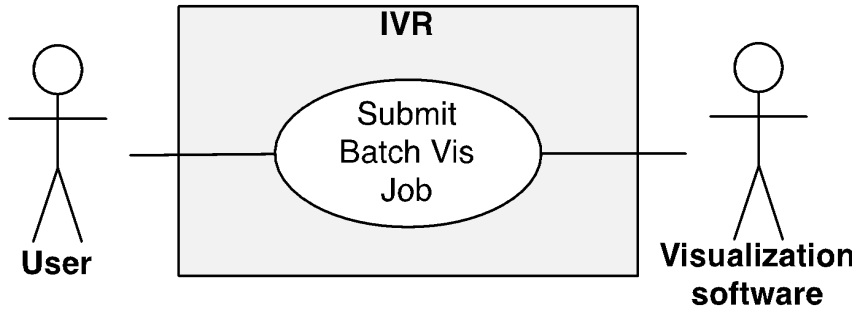




Visualization Use Case 2 Diagram

Use Case	**VISUALIZATION.3.0**
Remote, Batch Visualization	User interacting with a remote system and has need for visualization results in batch
References	***This is likely an extension of Architects Canonical Use Case 1 [citation]. The differences are:*** ***1. The appropriate visualization software is installed (note that as per the information use case appropriate meta data is properly loaded.)***
Actors	• XSEDE: SP Forum Members with visualization software installed • XSEDE: visualization teams • XSEDE: documentation and support teams • XSEDE User • Visualization software on the IVR

Prerequisites (Dependencies) and Assumptions	• Member of SP Forum with resources that have visualization software installed • Member of SP Forum that has data accessible by the resource • Visualization applications installed • Users have XSEDE authentication • Assumption: There is no need for high-performance interaction
Steps	• User identifies Resource (IVR) with visualization software installed • User installs any client side software required by the applications • User submits a batch job to the queue on the resource • The visualization software generates visualization products on the file system, where the user accesses them at a later time.
Variations (optional)	
Quality Attributes	Correct execution of the job.
Non-functional (optional)	
Issues	



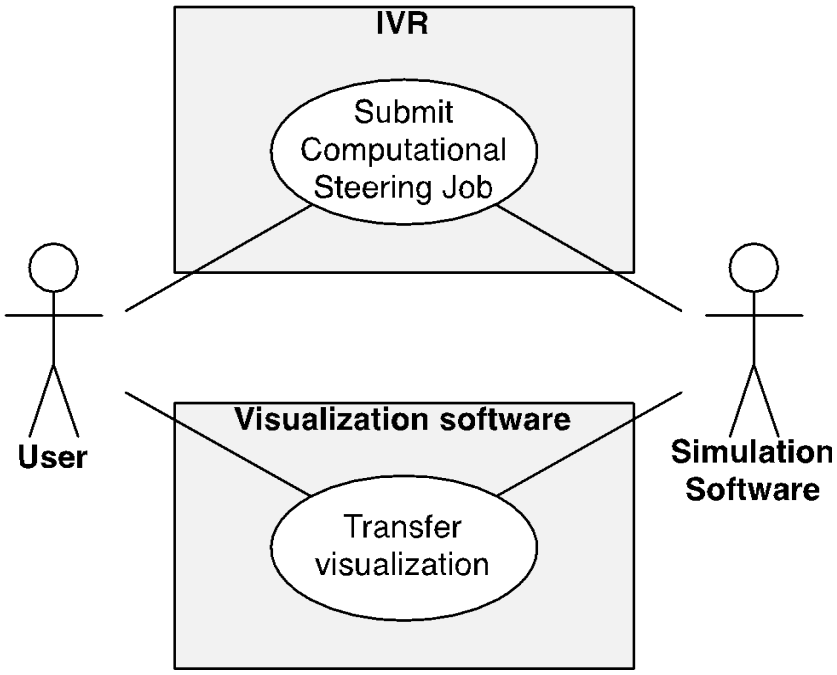
Visualization Use Case 3 Diagram

Use Case	**VISUALIZATION.4.0**
Computational Steering	User interacting with a remote system and has need for

	driving the simulation via interaction with the visualization.
References	***This incorporates use case 2 (above)*** ***This is likely an extension of Architects Canonical Use Case 1 [citation]. Mechanism to interact with the job is needed: e.g., file system, sockets, memory.*** ***The differences are:*** ***Similarities*** b) Remote data needs to be accessed during application execution. There are few turnkey tools for this. VisIt's libsim is one exception.
Actors	• XSEDE: SP Forum Members with software deployed that will facilitate computational steering • XSEDE: visualization teams • XSEDE: documentation and support teams • XSEDE User • Simulation software running on the compute resource • Client-side visualization software • Server-side steering software
Prerequisites (Dependencies) and Assumptions	• Member of SP Forum with software deployed that will facilitate computational steering. Computational steering is primarily a software issue and must be facilitated by both the simulation code and the visualization code • Member of SP Forum that has data accessible by resource with software capable of facilitating computational steering • Visualization applications to facilitate computational steering • User has XSEDE authentication and local machines that allow visualization. Machines with graphics capability are encouraged, but not necessary. • Network with sufficient bandwidth to provide performance of >= 10 frames/second • MUST support client software on a user-controlled laptop or desktop, perhaps behind a NAT or firewall.
Steps	• User identifies XSEDE resource with computational steering software • User submits a computational steering job to the batch queue

	• User connects to the running batch job through the computational steering software • User wants to fire up viz tool on local machine • Data is over on some other machine • I want to run the analysis there • Needs discovery capabilities, to find resources that can run the software (e.g., visit tool has machine profiles with relevant details, queues, etc. Every machine has its own profile. When you change profile you need to change it. And all of this is left as an exercise for the user.) • How to get jobs connected to each other, through firewalls (e.g., ssh forwarding -- but don't know what sockets will be available when doing the ssh, is there a better forwarding approach?) • Authentication • What queues are appropriate for use • How to submit to those queues • How to connect backend to frontend through established tunnels • "I want to run on 50 processors on this resource, go."
Variations (optional)	Communication variations: a) Via the file system b) Via sockets c) Via memory d) Named pipes "remote" means two pieces of software that do not have a direct form of communication - they do not share a file system, there are no direct sockets, there is no shared memory. They may not be in the same authentication realm. API so that two remote pieces of software can communicate in a two-way, secure (authenticated and encrypted), high performing (QoS definition needed), reliable way. With streams, messages (with significant payload). Object in one memory space "appears" in the memory space of another object. Single reader/writer. Would like multiple readers/writers. Need to define semantics on multiple readers (multicast versus races)

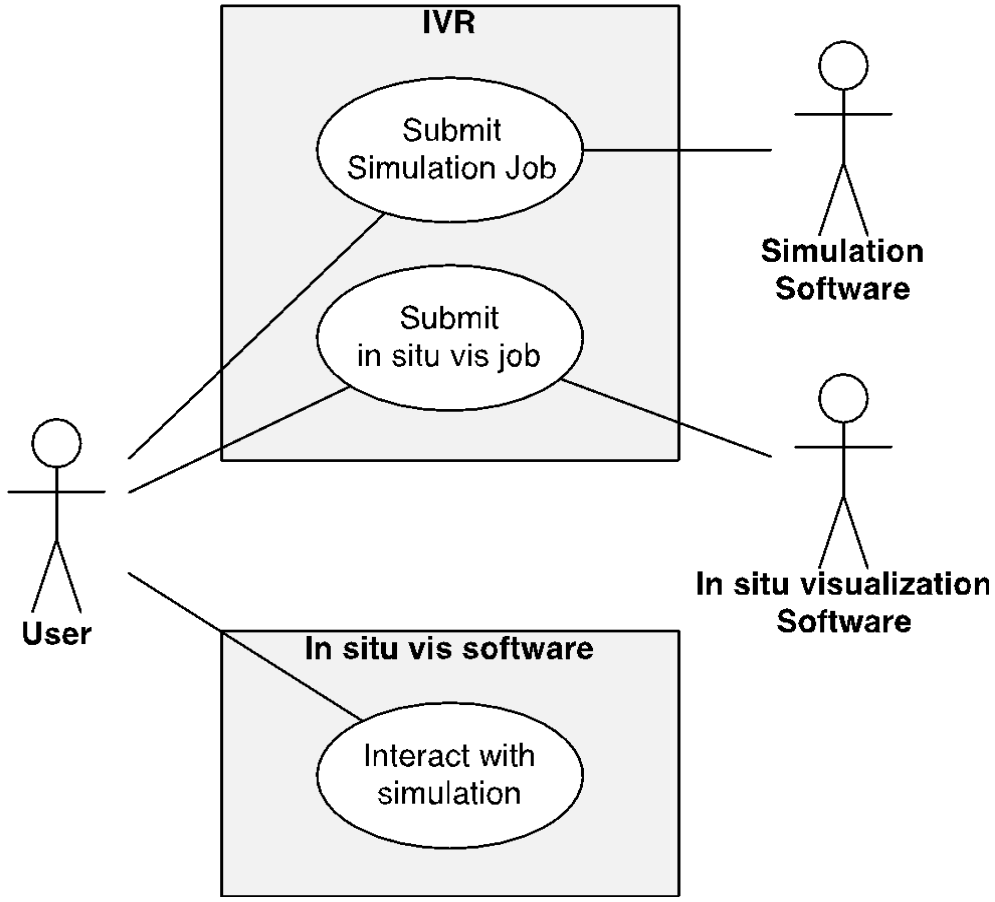
	Collaborative vis is something they'd like to do. open("path") : read/write on streams open("path"): send/receive messages (post,?)
Quality Attributes	• Provide interactive performance of >= 10 frames/second (if the analysis can produce images at this rate) • User must never be asked to authenticate more than once • Availability? • Client installation time? • Requirements to not install additional software on the client (beyond the viz tool)?
Non-functional (optional)	
Issues	•



Visualization Use Case 4 Diagram

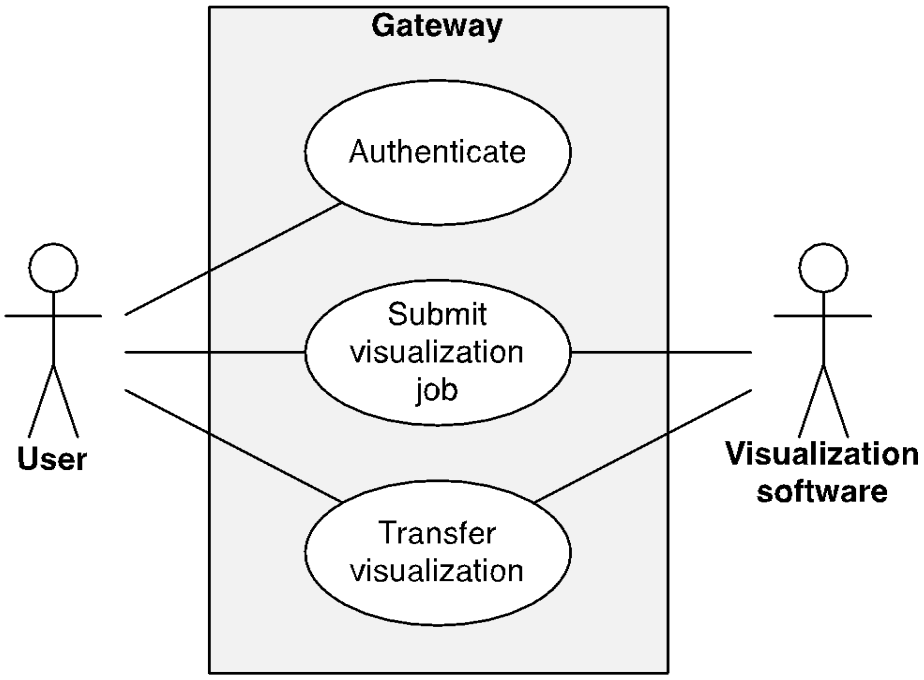
Use Case	**VISUALIZATION.5.0**
In Situ Visualization	User interacting with a remote system and has need for in situ visualization/analysis. (In situ means that visualization/analysis are part of the simulation, within the same memory space, usually for the purpose of monitoring and visualization. In general, this serves as the primary method of analysis of simulation data.)
References	
Actors	• XSEDE: SP Forum Members with software deployed that will facilitate in situ visualization • XSEDE: visualization teams • XSEDE: documentation and support teams • XSEDE User • Simulation software running on the compute resource • In situ visualization software running on the compute resource
Prerequisites (Dependencies) and Assumptions	• Member of SP Forum with software deployed that will facilitate in situ visualization. In situ is primarily a software issue and must be facilitated by both the simulation code and the visualization code • Member of SP Forum that has data accessible by resource with software capable of facilitating in situ visualization • Visualization applications to facilitate in situ visualization • Users with XSEDE authentication and local machines that allow visualization. Machines with graphics capability are encouraged, but not necessary.
Steps	• User identifies XSEDE resource with in situ visualization software • User submits an in situ visualization job to the batch queue • User connects to the in situ visualization job and then connects to a running simulation • Connecting to it with remote viewing software • Traverse firewalls • (But not controlling. Just viewing.)
Variations (optional)	

Quality Attributes	• Dependent on network to provide interactive performance of >= 10 frames/second
Non-functional (optional)	
Issues	•



Visualization Use Case 5 Diagram

Use Case	VISUALIZATION.6.0
Visualization Gateways	User interacting with a remote visualization system via a web interface (and steering and in situ?)
References	
Actors	• XSEDE: SP Forum Members with gateways deployed on XSEDE resources • XSEDE: visualization team • XSEDE: documentation and support teams • XSEDE User • Visualization software running on the visualization resource
Prerequisites (Dependencies) and Assumptions	• Member of SP Forum with gateway software to act as a front end to a visualization resource • Member of SP Forum that has data accessible by resource and gateway so that data can be read from the visualization resource when the job launches • Visualization gateway deployed to facilitate remote, interactive visualization • Users with XSEDE authentication and local machines that allow for remote, interactive visualization and have a gateway as a front end for remote, interactive visualization • Network with sufficient bandwidth to provide performance of >= 10 frames/second • Only software that user has on local machine is a standard web browser (flash?, java?, plug-ins?)
Steps	• User identifies visualization gateway which determines which resource jobs will be run on • User logs in to the gateway and authenticates • User submits a remote, interactive visualization job via the gateway • Visualization job launches in the gateway window • • Find proper web site • Point browser at web site • Authenticate through browser • ...
Variations (optional)	
Quality Attributes	• Dependent on network, provide interactive performance of >= 10 frames/second.
Non-functional (optional)	
Issues	•



Visualization Use Case 6 Diagram